Objektorientierte Programmierung

Kapitel 0: Organisatorisches

Prof. Dr. Stefan Brass

Martin-Luther-Universität Halle-Wittenberg

Wintersemester 2018/19

http://www.informatik.uni-halle.de/~brass/oop18/

Inhalt

Lernziele

- Programmieren können

  Selbständige Erstellung eigener Programme für gegebene Aufgaben.
  Nur kleinere/einfachere Programme, z.B. werden interessantere Algorithmen und Datenstrukturen hier nicht behandelt (eigene Vorlesung).

- Rechner, Betriebssystem und Editor bzw. Entwicklungsumgebung ausreichend bedienen können.

- Gegebene Programme verstehen können.

  Z.B. Klausurfrage: Was gibt dieses Programm aus?

- Die syntaktische Korrektheit gegebener Programme beurteilen können ($\rightarrow$ Syntax-Diagramme).

- Grundkonzepte von Programmiersprachen kennen, sich leicht in neue Sprachen einarbeiten können.

Zur Programmiersprache

- In dieser Vorlesung wird Programmierung anhand der Sprache "Java" gelehrt. Java ist:
  - In der Praxis verbreitet.
  - Gut mit Software-Werkzeugen und Literatur unterstützt.
- Es ist nicht sehr wichtig, mit welcher Sprache man beginnt.

  Der wesentliche Aufwand beim "Programmieren lernen" sind nicht so sehr die spezifischen Konstrukte einer Programmiersprache, als zu lernen, sich in Computer hineinzudenken, und einen Schatz von Mustern für bestimmte Aufgaben im Kopf zu haben, die man nach Bedarf aktivieren und anpassen kann. Diese Muster sind aber relativ unabhängig von der Programmiersprache, außer bei ganz anderen Programmierparadigmen, wie etwa logischer Programmierung. Viele Sprachen stammen wie Java aus der "C-Familie". Sie werden noch viele andere Programmiersprachen lernen müssen.

Inhalt

Ansprechpartner (1)

Dozent: Prof. Dr. Stefan Brass

- Email: brass@informatik.uni-halle.de

  Betreff-Zeile sollte mit [oop18] beginnen, möglichst aussagefähig.

  Falls keine Antwort: nochmal senden.

- Büro: Von-Seckendorff-Platz 1, Raum 313.
- Telefon: 0345/55-24740.
- Sprechstunde: Montags, 12^{00}–13^{00}.
- Frühere Unis: Braunschweig, Dortmund, Hannover, Hildesheim, Pittsburgh, Gießen, Clausthal.
- Spezialgebiet: Datenbanken, Wissensrepräsentation.

  Oracle8 Certified DBA. IBM Certified Advanced DBA (DB2 8.1).

Ansprechpartner (2)

Übung (Gruppen 2, 7, 8): Annett Thüring

- Büro: Von-Seckendorff-Platz 1, Raum 319.
- Telefon: 0345/55-24739.
- Email: thuering@informatik.uni-halle.de

Übung (Gruppen 1, 3): Mario Wenzel

- Büro: Von-Seckendorff-Platz 1, Raum 315.
- Telefon: 0345/55-24776.
- Email: mario.wenzel@informatik.uni-halle.de

Ansprechpartner (3)

Übung (Gruppen 4, 6): Vaibhav Kasturia

- Büro: Von-Seckendorff-Platz 1, Raum 211.
- Telefon: 0345/55-24718.
- Email: vaibhav.kasturia@informatik.uni-halle.de

Übung (Gruppe 5): Janek Bevendorff

- Büro: Von-Seckendorff-Platz 1, Raum ?.
- Telefon: 0345/55-24718.
- Email: janek.bevendorff@informatik.uni-halle.de

Ansprechpartner (4)

Sekretärin: Ramona Vahrenhold

- Büro: Von-Seckendorff-Platz 1, Raum 324.
- Telefon: 0345/55-24750, Fax: 0345/55-27333.
- Email: vahrenhold@informatik.uni-halle.de

Rechnerbetriebsgruppe:

- Poolaufsicht (studentische Hilfskräfte): Raum 318.
- Daniel Trull: Raum 320.

  Telefon: 0345/55-24717, EMail: daniel.trull@informatik.uni-halle.de
- Leiterin der Rechnerbetriebsgruppe ist Frau Thüring.

Webseiten

- http://www.informatik.uni-halle.de/~brass/oop18/
  - Folien (Kurzfassung in Vorlesung, Langfassung als Skript).

    Die Folien werden jeweils vor der Vorlesung ins Web gestellt. Es gibt auch eine erweiterte Fassung, u.a. mit selten benötigten Java-Konstrukten und Vergleichen zu C++. Diese Zusätze sind nicht prüfungsrelevant.
  - Alte Klausuren und Programmiertests.
  - Nützliche Links.
- StudIP: https://studip.uni-halle.de/dispatch.php/course/details?sem_id=5cf5e121ed5d8e5d635855734479bab
  - Anmeldung zu Vorlesung und Übungsgruppen.
  - Link zu Übungsplattform (Hausaufgaben).
  - Forum (z.B. auch Fragen zu Hausaufgaben).

Zeit und Ort (1)

Vorlesung (2 SWS):

- Dienstags, 10^{15}–11^{45}, Raum 3.28.

  302 Teilnehmer in StudIP, Raum hat 196 Plätze. Z.B. nochmal Di 18–20?

Übung am Rechner (2 SWS):

- Acht Gruppen, je max. 27 Teilnehmer, Beginn 22./23.10.:

Nr	Tag	Zeit	Raum	Leiter	Teiln.
1/8	Montag	12^{15}–13^{45}	3.34/3.02	MW/AT	71
2	Montag	14^{15}–15^{45}	3.34	AT	47
3	Montag	16^{15}–17^{45}	3.34	MW	43
4/5	Dienstag	12^{15}–13^{45}	3.34/3.02	VK/JB	85
6/7	Dienstag	14^{15}–15^{45}	3.34/3.02	VK/AT	62

9. Gruppe in Vorbereitung. Freiwillige Umverteilung. StudIP-Link unter Nr.

Zeit und Ort (2)

Anmeldung zu Übungsgruppen:

- Tragen Sie sich in StudIP für eine Gruppe ein.

  [http://studip.uni-halle.de/]. Direkte Links auf der Vorlesungs-Webseite:
  [http://www.informatik.uni-halle.de/~brass/oop18/termine.html].
  Falls noch kein StudIP-Zugang: Papierliste hier.

- Wir garantieren, dass Sie einen Platz in einer der Übungsgruppen bekommen.

  Wenn Sie zur Teilnahme an dieser Vorlesung verpflichtet sind.

- Wir können leider nicht garantieren, dass Sie einen Platz in einer bestimmten Gruppe bekommen.

  Bitte wechseln Sie möglichst aus den überfüllten Gruppen in eine andere.
  Wenn sich das Problem nicht freiwillig löst, werden wir umverteilen.

Zeit und Ort (3)

Anmeldung zu Übungsgruppen, Forts.:

- Für die Anmeldung zu einer Übungsgruppe über StudIP brauchen Sie einen Benutzernamen und ein Passwort, das Ihnen mit der Immatrikulationsbescheinigung zugegangen sein müsste.

  Der Benutzername identifiziert Sie eindeutig im System (Vor- und Nachname sind nicht immer eindeutig). Das Passwort sollten nur Sie wissen.
  Es dient als Ausweis, dass Sie auch wirklich Sie sind.

- Bitte tragen Sie sich bei StudIP sowohl
  - für die Übungsgruppe Ihrer Wahl, als auch
  - für die Vorlesung selbst ein.

    Gelegentlich werden wichtige Mitteilungen über diesen Verteiler versendet.

Linux

- Die Rechner in den Pools laufen unter Linux.

  Eventuell haben sie eine "Dual Boot" Möglichkeit (Windows oder Linux), aber es ist für uns einfacher, wenn alle in den Übungen Linux verwenden.

- Es ist zu empfehlen, Linux zu lernen.

  Früher oder später brauchen Sie es in Ihrem Studium (und im Beruf).
  Die Horizonterweiterung ("nicht alles ist Windows!") ist wichtig und hilfreich.
  Linux ist "open source" und man kann im Prinzip alle Interna anschauen.

- Die Programmierung in den Übungen geschieht über eine Webschnittstelle (YAPEX, mit automatischen Tests), Sie brauchen also sehr wenig Linux.

  Zu Hause können Sie natürlich unter Windows arbeiten. Die Hausaufgaben müssen über YAPEX angegeben werden, aber sie können die Programme vorher mit anderen Werkzeugen entwickeln und ausprobieren.

Tutorium / OOP-Sprechstunde

- Wenn Sie Hilfe brauchen, bieten wir folgenden optionalen Termin:

Tag	Zeit	Raum
Donnerstag	12^{15}–13^{45}	3.34

- Es sind hier zwei Tutoren anwesend, also Studierende in höheren Semestern.
- Sie können beliebige Fragen zu OOP stellen.
- Da es im Rechnerpool ist, ggf. auch mit mehr Hilfe üben.
- Das Tutorium ist nur gedacht für Studierende, die mit der Programmierung Schwierigkeiten haben.

Anmeldung: Übersicht / Checkliste

Um dieses Modul erfolgreich abzuschließen, müssen Sie

- ☐ sich bei StudIP für eine Übungsgruppe anmelden,
- ☐ sich zum Modul anmelden (Löwenportal, bis ca. 28.10.2018),
- ☐ die Vorlesung nacharbeiten (Skript, Buch, ausprobieren),
- ☐ Hausaufgaben bearbeiten und abgeben,
  und dabei 50% der "Theoriepunkte" sammeln,
- ☐ in den Übungen ausreichend aktiv mitarbeiten,
  und dabei durch Lösen praktischer Programmieraufgaben am Rechner mindestens 50% der "Praxispunkte" bekommen,
- ☐ sich zur Prüfung anmelden (Löwenportal),
- ☐ die Prüfung bestehen (voraussichtlich 26.02./26.03.2019).

Inhalt

Wichtig: Vorlesung nacharbeiten (1)

- Obwohl es nicht explizit auf dem Aufgabenblatt steht, ist jede Woche eine Hausaufgabe, die Vorlesung nachzuarbeiten:
  - Lesen Sie die ausführliche Fassung der Folien.
  - Gehen Sie dabei auch Ihre Aufzeichnungen aus der Vorlesung nochmal durch, notieren Sie sich Fragen.

    Und sorgen Sie für die Klärung dieser Fragen. Z.B. in nächster Vorlesung, Übung oder Forum stellen. Oder selbst ausprobieren. Im Internet suchen.
  - Lesen Sie zusätzliche Literatur: Lehrbücher, Internet-Quellen.
  - Probieren Sie kritische Dinge am Rechner aus.
  - Schreiben Sie sich mit eigenen Worten das Wichtigste auf.

    Am Ende eigene "Quick Reference". Bei der Klausur sind 5 Blätter erlaubt.
  - Reden Sie mit anderen Studierenden (bilden Sie Lerngruppen).

Wichtig: Vorlesung nacharbeiten (2)

- Wenn Sie die Vorlesung nicht nacharbeiten (und nicht bereits über gute Java-Programmierkenntnisse verfügen), werden Sie in dieser Vorlesung untergehen.

  In der nächsten Woche kommt schon wieder neuer Stoff.

- In der Modulbeschreibung stehen 90 Stunden Arbeitszeit (im Semester) für Hausaufgaben und Selbststudium.

  Durchschnittliche Arbeitszeit: Studenten ohne Vorkenntnisse eher mehr, mit Vorkenntnissen eher weniger (hängt aber auch von anderen Faktoren ab).

  - Z.B. 2 Stunden pro Woche für Hausaufgaben.

    Wenn Sie wesentlich mehr brauchen, stimmt etwas nicht.

  - Z.B. 4 Stunden pro Woche zur Nacharbeit.

    Ggf. müssen Sie aus Zeitgründen einen Teil auf die Semesterpause verschieben (Klausurvorbereitung), aber nicht so viel, das Sie im Semester abgehängt werden.

Statistik (1)

Note	2012/13		2013/14		2. Termin	
1.0	16	(14%)	22	(24%)	1	(6%)
1.3	13	(12%)	13	(14%)	1	(6%)
1.7	5	(4%)	12	(13%)	0	(0%)
2.0	8	(7%)	8	(9%)	4	(22%)
2.3	12	(11%)	6	(7%)	1	(6%)
2.7	12	(11%)	4	(4%)	0	(0%)
3.0	7	(6%)	4	(4%)	3	(17%)
3.3	7	(6%)	1	(1%)	1	(6%)
3.7	6	(5%)	0	(0%)	0	(0%)
4.0	7	(6%)	0	(0%)	0	(0%)
5.0	19	(17%)	21	(23%)	7	(39%)

2013/14 haben sich nur 51% der in StudIP für das Modul eingetragenen Studenten zur ersten Klausur angemeldet (2012/13: 71%).

Statistik (2)

- 2013/14 haben 7 der Teilnehmer, die bei der ersten Klausur durchgefallen waren, an der zweiten erneut teilgenommen.

  Davon ist einer wieder durchgefallen, zwei hatten eine 2.0.

- Bei der Klausur 2013/14 haben wir nach Vorkenntnissen ("Programmierkenntnisse vor der Vorlesung") gefragt:
  - Keine Vorkenntnisse: 34%

    Durchschnittsnote 2.3, sehr gut (1.0/1.3): 22%, nicht bestanden: 11%.
  - Etwas Vorkenntnisse: 41%

    Durchschnittsnote 2.4, sehr gut (1.0/1.3): 42%, nicht bestanden: 24%.

    Verteilung stark an den beiden Enden: Manche haben aus dem Vorwissen etwas gemacht, andere haben sich zu sehr darauf verlassen.
  - Viel Vorkenntnisse: 12%

    Durchschnittsnote 1.1, sehr gut (1.0/1.3): 100%, nicht bestanden: 0%.

Geben Sie nicht auf!

- Die bisherige Erfahrung zeigt, dass ein nicht unwesentlicher Teil der Studierenden, die diese Vorlesung anfangen, nicht bis zum Ende dabei bleibt (auch nicht zur Klausur kommt).
- Zum Teil lag es am Programmiertest (jetzt verändert).

  Sie bekommen früher und häufiger sanfteres Feedback, mit weniger Druck.

  Spätestens in der Klausur werden aber die gleichen Fähigkeiten verlangt.

  Tatsächlich war der Programmiertest ein guter Indikator für den Klausur-Erfolg.
- Zum Teil merken die Studierenden aber zu spät, dass sie abgehängt sind, oder es zu viel Arbeit ist.

  Studium in der Regelstudienzeit verlangt eher mehr als 40 Stunden/Woche (zumindest während der Vorlesungszeit).
- Bedenken Sie Modul-Abhängigkeiten.

  Sie können z.B. “Datenstrukturen und effiziente Algorithmen I” im 2. Semester nur besuchen, wenn Sie diese Vorlesung erfolgreich abgeschlossen haben.

Inhalt

Rechnernutzung, Software (1)

Rechnerpools:

- PC-Pool: Raum 3.34
- Multimedia-Pool: Raum 3.02
- Siehe: [http://www.informatik.uni-halle.de/studium/pools/]

  Sie können die Pools auch außerhalb Ihrer Übungszeiten nutzen, wenn sie nicht belegt sind (falls dort eine kleine Übung läuft, die nicht alle Rechner braucht, können Sie zumindest still auf der Empore arbeiten).

- Damit Sie auf den Rechnern in den Pools 3.34/3.02 arbeiten können, muss ein Benutzerkonto eingerichtet werden.

  Die Zugangsdaten entsprechen zu Beginn des Studiums dem Account, welchen Sie mit Ihren Immatrikulationsunterlagen erhalten haben.
  Bitte prüfen Sie vor der ersten Übung, ob Sie sich mit Ihren Zugangsdaten in den Pools anmelden können. Falls eine Anmeldung nicht möglich ist, so melden Sie sich bitte bei der Rechnerbetriebsgruppe (Räume: 318, 319, 320).

Rechnernutzung, Software (2)

Falls Sie einen eigenen PC haben:

- Sie brauchen mindestens das JDK ("Java Development Kit") für Java SE 8 von Sun/Oracle:
  [http://www.oracle.com/technetwork/java/javase/downloads/]

  Sun hat Java entwickelt, wurde aber inzwischen von Oracle gekauft.
  SE steht für "Standard Edition". Die Unterschiede zwischen den Versionen 6 bis 11 der Sprache sind für diese Vorlesung nicht wichtig. Java 8 wird noch unterstützt, Java 10 ist aktuelle "Feature Release", Java 11 ist erste "Long Term Support" Release (das nur frei zur Entwicklung, oder GPL).
  Das JDK enthält das JRE (Java Runtime Environment), das zur Ausführung von Java-Programmen nötig ist. Das JRE ist auf vielen Rechnern schon installiert (spätestens mit dem ersten Java-Programm). Zur Programmentwicklung brauchen Sie die zusätzlichen Bestandteile des JDK (insbesondere den Compiler).
- Entwicklungsumgebungen (z.B. Eclipse, Netbeans, BlueJ) enthalten dies oft schon.

Beschaffen Sie sich ein Lehrbuch!

- Es gibt verschiedene Lerntypen.
  Meine Vorlesung ist nicht für jeden optimal.

  Mir ist Präzision wichtig. Ich versuche alles systematisch zu definieren, inklusive der aus meiner Sicht nötigen Details. Manche Aussagen wären ohne die Details strenggenommen falsch. Im Laufe der Zeit habe ich auch viele Missverständnisse und mögliche Fehler gesehen, und häufig mein Skript erweitert, um den jeweiligen Punkt noch genauer zu klären (so wird es lang).

- Am Ende ist die Hauptsache, dass Sie programmieren können.

  Wenn Sie eine "1" wollen, sind allerdings auch Details wichtig.

- Sie können sich die Kenntnisse auch auf anderem Weg aneignen, z.B. aus einem Buch, das Ihrem Lerntyp entspricht.

- Außerdem können Sie selbstbewusster und kritischer mitdenken, wenn Sie auch andere Quellen haben.

Internet-Kurse

- Es gibt im Internet viele Java-Kurse.
  - Wenn man z.B. bei Google "Video Tutorial Java" eingibt, erhält man 1.7 Millionen Webseiten.
  - Es gibt aufgezeichnete Vorlesungen anderer Professoren.
  - Es gibt interaktive Tutorials (z.B. mit Ankreuzaufgaben).
- Eine Auswahl steht auch unter [http://users.informatik.uni-halle.de/~brass/oop18/links.html]
- Fragen Sie mich nicht, welches das beste Tutorial ist.

  Ich halte naturgemäß diese Vorlesung für die beste Möglichkeit, Java zu lernen.
  Wenn Sie gute Tutorials finden, stelle ich sie gerne auf meine Webseite.
  Wenn Sie Tutorials auf meiner Seite für schlecht halten, teilen Sie mir das mit.

Inhalt

Vorkenntnisse (1)

- Diese Vorlesung wird von Studierenden mit sehr unterschiedlichen Vorkenntnissen besucht.
  - Einige können schon programmieren, ggf. sogar in Java.

    Bedenken Sie aber bitte, dass diese Vorlesung auch Konzepte vermitteln soll, die in einer rein praktischen Ausbildung fehlen.

    Achtung: Man kann den Punkt verpassen, wo es dann doch neu wird.
  - Für andere Teilnehmer ist die Programmierung ein völliges Neuland.

    Wir haben leider keine Lehrkapazität für zwei getrennte Vorlesungen.

    Es ist auch nicht klar, ob das wirklich helfen würde: Vorkenntnisse und Aufnahmegeschwindigkeit bleiben unterschiedlich.
- Mein Anspruch ist, dass auch die zweite Gruppe der Vorlesung folgen kann: Ich möchte den Stoff "von Grund auf" logisch und vollständig (aber zügig) präsentieren.

Vorkenntnisse (2)

- Stellen Sie Fragen, wenn Sie etwas nicht verstehen, gerne auch während der Vorlesung.

  Wer schon so lange programmiert wie ich, übersieht manchmal, dass bestimmte Dinge nicht selbstverständlich sind. Ich bin dankbar für Hinweise, die es mir erlauben, mein Skript (Foliensatz) noch zu verbessern. Andere Studierende werden Ihnen auch dankbar sein.

- Lassen Sie sich nicht einschüchtern, wenn andere mehr als Sie wissen.

  Offiziell fordert diese Vorlesung keine Vorkenntnisse, es ist also ok, wenn Sie keine haben. Es ist Ihr gutes Recht, zu fragen. Es ist auch völlig unklar, wie es gegen Ende der Vorlesung mit dem Wissen aussieht: Manche Programmier-Neulinge können sich gut in Maschinen und Formalismen hineindenken und lernen schnell. Mancher Teilnehmer mit Vorwissen hat über viele Details noch nie nachgedacht und wüßte die auch nicht.

Theorie vs. Praxis (1)

- Praktische Fähigkeiten wie die Programmierung erwirbt man neben dem
  - notwendigen theoretischen Wissen auch durch

    Präzision des Denkens, die sich aus theoretischem Wissen speist, ist gerade in der Programmierung wichtig.
  - praktisches Tun und Lernen aus Fehlern.
- Oft können auch Lösungen, die im Prinzip korrekt sind (funktionieren), noch verbessert werden.

  Z.B. einfacher, kürzer, übersichtlicher, leichter änderbar, resourcen-sparender (schneller/weniger CPU-Zeit oder weniger Speicherplatz). Nutzen Sie die Übung zum Austausch mit anderen Studierenden und dem Tutor (natürlich möglichst nach Abgabe der Hausaufgaben).
- Wie viel Sie lernen, hängt an der investierten Zeit.

Theorie vs. Praxis (2)

- Wenn praktische Fähigkeiten auch notwendig sind, ist dies doch eine Vorlesung an einer Universität:
  - Man darf über Java auch kritisch nachdenken.

    Mir gefällt auch nicht alles in Java. Überlegen Sie sich, was Sie als Programmiersprachen-Entwerfer anders machen würden.
  - Ziel ist auch ein möglichst leichter Übergang zu anderen Sprachen (allgemeine Konzepte!).
  - Auswendiglernen ("pauken") sollte nicht nötig sein.

    Wichtige Dinge prägen sich bei ausreichender theoretischer und praktischer Beschäftigung mit der Programmierung automatisch ein, selten benötigte Details kann man bei Bedarf nachschlagen. Das meiste ist logisch aufgebaut, und man kann die Gründe dafür verstehen.
  - Ihre Mitwirkung (Fragen, Vorschläge) ist wichtig!

Vorlesungs-Etikette

- Vermeiden Sie Verhalten, das Ihre Mitstudenten oder den Professor ablenkt:
  - Vermeiden Sie Gespräche während der Vorlesung.

    Glauben Sie nicht, dass Sie in der Masse untergehen: Der Professor kann durchaus sehen, wer sich unterhält. Wenn Sie Ihren Nachbarn etwas zur Vorlesung fragen müssen, machen Sie es leise und kurz. Wenn die Frage möglicherweise auch für andere interessant ist, stellen Sie sie offiziell (melden, ggf. rufen).
  - Wenn Sie zu spät kommen oder früher gehen müssen, setzen Sie sich möglichst an den Rand.
  - Notebooks sollten während der Vorlesung nur die Folien anzeigen (eventuell Editor, Compiler).
- Nur anwesend zu sein, reicht nicht. Sie müssen mitdenken.

  Und ggf. Fragen stellen. Sie sind erwachsen. Verschwenden Sie Ihre Zeit nicht.

Ratschläge zum Studium (1)

- Professoren freuen sich über Fragen!
- Scheuen Sie sich nicht vor anderen Studierenden, die den Eindruck machen, schon mehr zu wissen.

  Gegen Ende des Semesters kann es schon ganz anders aussehen. Es ist keine Schande, Informatik nicht schon in der Schule gehabt zu haben.
- Lesen Sie ein Buch zum Thema der Vorlesung, nicht nur das Skript. Benutzen Sie unterschiedliche Quellen.

  Seien Sie selbständig und etwas kritisch. Glauben Sie nicht etwas, nur weil Sie es zufällig gehört haben. Versuchen Sie zu verstehen, nicht auswendig zu lernen.
- Nehmen Sie das Studium ernst.

  Es gibt mehr Freiheiten als an der Schule, und die negativen Auswirkungen eines Missbrauchs dieser Freiheit zeigt sich erst mit etwas Verzögerung.
  Auch hier gilt: Ohne Fleiss kein Preis. Soviel Zeit, wie Sie jetzt haben, sich fortzubilden und Bücher zu lesen, haben Sie später vermutlich nie wieder.

Ratschläge zum Studium (2)

- Lernen Sie programmieren (und wirklich gut).
  Es ist das Handwerk, auf dem alles beruht.
- Lernen Sie mathematisches Denken (Formalisieren, Beweisen), und Techniken wie z.B. vollständige Induktion, Grundlagen der Algebra und Logik.
- Arbeiten Sie in Gruppen zusammen, aber sorgen Sie dafür, dass jedes Gruppenmitglied alles lernt.
- Falls Frage nach Sinn des Lebens: Ich bin Christ.

  Das bedeutet nicht, das jeder am Ende eine 1.0 bekommt. Das wäre Untreue meinem Arbeitgeber gegenüber und würde auch Ihnen letztendlich wenig nützen. Sie können aber davon ausgehen, dass mir wichtig ist, dass Sie etwas lernen, und ich Ihnen helfen will. Leider sind meine Zeit und Fähigkeiten begrenzt.